

# XML4C



# AND

## Standardizing Configuration Input for Visual Simulation

LT Ken Miller

Capt Steve Matthews

# Motivation

- Our thesis is avatar (human) visual simulation. To be dynamic, we need to be able to load different models that follow the same pattern (the same information and types of model files)
- Our use of XML and XML4C was driven by the desire to provide a common format for the configuration file passed into our simulation.



## XML4C Version 4.0.1

XML4C is a validating XML parser written in a portable subset of C++. XML4C makes it easy to give your application the ability to read and write XML data. A shared library is provided for parsing, generating, manipulating, and validating XML documents.

XML4C is faithful to the XML 1.0 recommendation and associated standards ( DOM 1.0, DOM 2.0, SAX 1.0, SAX 2.0, Namespaces, and W3C's XML Schema recommendation version 1.0.)

The parser provides high performance, modularity, and scalability. Source code, samples and API documentation are provided with the parser. For portability, care has been taken to make minimal use of templates, no RTTI, no C++ namespaces and minimal use of #ifdefs.



## Applications of the XML4C Parser

XML4C has rich generating and validating capabilities. The parser is used for:

- Building XML-savvy Web servers
- Building next generation of vertical applications that use XML as their data format
- On-the-fly validation for creating XML editors
- Ensuring the integrity of e-business data expressed in XML
- Building truly internationalized XML applications



## Features

- Conforms to XML Spec 1.0
- Tracking of latest DOM (Level 1.0), DOM (Level 2.0), SAX/SAX2, Namespace, and W3C's XML Schema recommendation version 1.0 specifications.
- Source code, samples, and documentation is provided.
- Programmatic generation and validation of XML
- Pluggable catalogs, validators and encodings
- High performance
- Customizable error handling



## API Docs for SAX and DOM

XML4C is packaged with the API documentation for SAX and DOM, the two most common programming interfaces for XML. The most common framework classes have also been documented.

XML4C DOM is an implementation of the [Document Object Model \(Core\) Level 1](#) as defined in the W3C Recommendation of 1 October, 1998; and [Document Object Model \(Core\) Level 2](#) as defined in the W3C Recommendation of 13 November, 2000. For a complete understanding of how the XML4C APIs work, we recommend you to read these documents.

XML4C SAX is an implementation of the [SAX 1.0/2.0](#) specification. You are encouraged to read this document for a better understanding of the SAX API in XML4C.



# XML for C++ Parser

## What compilers are being used on the supported platforms?

XML4C binaries has been built on the following platforms with these compilers

Operating System	Compiler
Windows NT 4.0 SP5/98	MSVC 6.0 SP3
Redhat Linux 6.1	egcs-2.91.66 and glibc-2.1.2-11
AIX 4.3	xlC_r 5.0.2
Solaris 2.6	Forte C++ Version 6 Update 2
HP-UX 11.0	aCC A.03.13 with pthreads



# XML for C++ Parser

## What are the differences between Xerces-C and XML4C?

Xerces-C has intrinsic support for ASCII, UTF-8, UTF-16 (Big/Small Endian), UCS4 (Big/Small Endian), EBCDIC code pages IBM037 and IBM1140 encodings, ISO-8859-1 (aka Latin1) and Windows-1252. This means that it can parse input XML files in these above mentioned encodings.

However, if you wish to parse XML files in any other encodings, say in Shift-JIS, Big5 etc., then you cannot use Xerces-C. XML4C addresses this need. It combines Xerces-C and [International Components for Unicode \(ICU\)](#) and provides support for over 100 different encodings.

ICU is also an open source project but is licensed under the [X License](#). XML4C is published by IBM and can be downloaded from their [Alphaworks](#) site. The license to use XML4C is simply to comply with the Apache license (because of Xerces-C) and X License (because of ICU).

XML4C binaries are published for Solaris using SunWorkshop compiler, HP-UX 10.20 and 11.0 using CC and aCC, Redhat Linux using gcc, Windows NT using MSVC, AIX using xlc.



# XML for C++ Parser

## XML4C Class Hierarchy

- AttributeList
- Attributes
- Base64
- BinInputStream
  - BinFileInputStream
  - BinMemInputStream
- ChildNode
  - ParentNode
- ContentHandler
  - DefaultHandler
- DocTypeHandler
  - DOMParser
  - IDOMParser
  - SAXParser
- DocumentHandler
  - HandlerBase
- DOM\_DOMException
  - DOM\_RangeException
- DOM\_DOMImplementation
- DOM\_NamedNodeMap
- DOM\_Node
  - DOM\_Attr
  - DOM\_CharacterData
    - DOM\_Comment
    - DOM\_Text
    - DOM\_CDATASection
  - DOM\_Document
  - DOM\_DocumentFragment
  - DOM\_DocumentType
  - DOM\_Element
  - DOM\_Entity
  - DOM\_EntityReference
  - DOM\_Notation
  - DOM\_ProcessingInstruction
  - DOM\_XMLDecl
- DOM\_NodeFilter
- DOM\_NodeIterator
- DOM\_NodeList
- DOM\_Range
- DOM\_TreeWalker
- DOMString
- DTDHandler
  - DefaultHandler
  - HandlerBase
- EntityResolver
  - DefaultHandler
  - HandlerBase
- ErrorHandler
  - DefaultHandler
  - HandlerBase
- HashBase
  - HashCMStateSet
- HexBin
- InputSource
  - LocalFileInputSource
  - MemBufInputSource
  - StdInInputSource
  - URLInputSource
- LexicalHandler
  - DefaultHandler
- Locator
- Parser
  - SAXParser
- QName
- SAX2XMLReader
- SAXException
  - SAXNotRecognizedException
  - SAXNotSupportedException
  - SAXParseException
- XMLTransService::TransRec
- XMLAttDef
- XMLAttDefList
- XMLAttr
- XMLBigInteger
- XMLContentModel
- XMLDeleter
  - XMLDeleterFor
- XMLDocumentHandler
  - DOMParser
  - IDOMParser
  - SAXParser
- XMLElementDecl
- XMLEntityDecl
- XMLEntityHandler
  - DOMParser
  - IDOMParser
  - SAXParser
- XMLErrorHandler
  - DOMParser
  - IDOMParser
  - SAXParser
- XMLErrs
- XMLException
  - NoDefTranscoderException
- XMLFormatTarget
- XMLFormatter
- XMLInteger
- XMLLCPTranscoder
- XMLNotationDecl
- XMLNumber
  - XMLAbstractDoubleFloat
    - XMLDouble
    - XMLFloat
  - XMLBigDecimal
  - XMLDateTime
- XMLPlatformUtils
- XMLReaderFactory
- XMLRegisterCleanup
- XMLString
- XMLStringTokenizer
- XMLTranscoder
- XMLTransService
- XMLUni
- XMLUri
- XMLValid
- XMLValidator



# XML for C++ Parser

## XML4C Samples

- [SAXCount](#)  
SAXCount counts the elements, attributes, spaces and characters in an XML file.
- [SAXPrint](#)  
SAXPrint parses an XML file and prints it out.
- [DOMCount](#)  
DOMCount counts the elements in a XML file.
- [DOMPrint](#)  
DOMPrint parses an XML file and prints it out.
- [MemParse](#)  
MemParse parses XML in a memory buffer, outputting the number of elements and attributes.
- [Redirect](#)  
Redirect redirects the input stream for external entities.
- [PParse](#)  
PParse demonstrates progressive parsing.
- [StdInParse](#)  
StdInParse demonstrates streaming XML data from standard input.
- [EnumVal](#)  
EnumVal shows how to enumerate the markup decls in a DTD Grammar.
- [SEnumVal](#)  
SEnumVal shows how to enumerate the markup decls in a Schema Grammar.
- [CreateDOMDocument](#)  
CreateDOMDocument creates a DOM tree in memory from scratch.
- [SAX2Count](#)  
SAX2Count counts the elements, attributes, spaces and characters in an XML file.
- [SAX2Print](#)  
SAX2Print parses an XML file and prints it out.
- [IDOMCount](#)  
IDOMCount counts the elements in a XML file.
- [IDOMPrint](#)  
IDOMPrint parses an XML file and prints it out.

# What We Found

- A critical look : XML4C has some issues.
  - XML4C stores in UTF encoding, which is not compatible with C++ character format; we had to translate each time we received a value.
  - Lack of reliable technical support. Did not receive prompt answer to questions concerning format and syntax, and the answers we received were much like the previous slides, “...go read...”

# What We Found

- XML4C provides for simpler code in the simulation model code.
  - Brute force c/c++ string parsing works fine, but is a mess to code and to maintain.
  - XML4C provides simpler tokenization of data, making the code in our project cleaner.

# What We Found

## ● The drawbacks:

- There is a distinct learning curve to XML4C, and this is significant not just for us, but for future code maintenance.
- XML4C is another API to maintain. This becomes less trivial as the project gets larger. We are currently working with several APIs and using Current Version Software (CVS) to maintain code, so the use of an additional API is a project decision.

# Comparison : The Config File

```
#  
# cal3d model configuration file  
#  
# model: cally  
#  
  
path=cally/  
  
scale=0.008  
  
skeleton=cally.csf  
  
state=idle 12 cally_idle.caf  
state=walk 20 cally_walk.caf  
state=strut 30 cally_strut.caf  
state=run 30 cally_jog.caf  
action=jumpkick cally_tornado_kick.caf  
action=shootarrow cally_shoot_arrow.caf  
action=wave cally_wave.caf  
blendedstate=runwave 20 run 0.3 wave 0.7
```

```
mesh=cally_calf_left.cmf  
mesh=cally_calf_right.cmf  
mesh=cally_chest.cmf  
mesh=cally_foot_left.cmf  
mesh=cally_foot_right.cmf  
mesh=cally_hand_left.cmf  
mesh=cally_hand_right.cmf  
mesh=cally_head.cmf  
mesh=cally_lowerarm_left.cmf  
mesh=cally_lowerarm_right.cmf  
mesh=cally_neck.cmf  
mesh=cally_pelvis.cmf  
mesh=cally_ponytail.cmf  
mesh=cally_thigh_left.cmf  
mesh=cally_thigh_right.cmf  
mesh=cally_upperarm_left.cmf  
mesh=cally_upperarm_right.cmf
```

```
material=cally_skin.crf  
material=cally_ponytail.crf  
material=cally_chest.crf  
material=cally_pelvis.crf
```

# Comparison : The Config File

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v4.3 U (http://www.xmlspy.com) by Ken Miller (Naval Postgraduate School) -->
<!--Sample XML file generated by XML Spy v4.3 U (http://www.xmlspy.com)-->
<model xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation=".\\model.xsd" path="cally\\" scale="0.008" skeleton="cally.csf">
    <animation>
        <type>state</type>
        <animation_name>Idle</animation_name>
        <transition>11</transition>
        <file>cally_idle.caf</file>
    </animation>
    <animation>
        <type>state</type>
        <animation_name>walk</animation_name>
        <transition>50</transition>
        <file>cally_walk.caf</file>
    </animation>
    <animation>
        <type>state</type>
        <animation_name>strut</animation_name>
        <transition>50</transition>
        <file>cally_strut.caf</file>
    </animation>
    <animation>
        <type>state</type>
        <animation_name>run</animation_name>
        <transition>50</transition>
        <file>cally_jog.caf</file>
    </animation>
```

# Comparison : The Config File

```
<animation>
  <type>action</type>
  <animation_name>jumpkick</animation_name>
  <file>cally_tornado_kick.caf</file>
</animation>
<animation>
  <type>action</type>
  <animation_name>shootarrow</animation_name>
  <file>cally_shoot_arrow.caf</file>
</animation>
<animation>
  <type>action</type>
  <animation_name>wave</animation_name>
  <file>cally_wave.caf</file>
</animation>
<animation>
  <type>blendedstate</type>
  <animation_name>runwave</animation_name>
  <transition>50</transition>
  <blend_animation_name>run</blend_animation_name>
  <blend_animation_factor>0.3</blend_animation_factor>
  <blend_animation_name>wave</blend_animation_name>
  <blend_animation_factor>0.7</blend_animation_factor>
</animation>
```

# Comparison : The Config File

<mesh>

```
<file>cally_calf_left.cmf</file>
<file>cally_calf_right.cmf</file>
<file>cally_chest.cmf</file>
<file>cally_foot_left.cmf</file>
<file>cally_foot_right.cmf</file>
<file>cally_hand_left.cmf</file>
<file>cally_hand_right.cmf</file>
<file>cally_head.cmf</file>
<file>cally_lowerarm_left.cmf</file>
<file>cally_lowerarm_right.cmf</file>
<file>cally_neck.cmf</file>
<file>cally_pelvis.cmf</file>
<file>cally_ponytail.cmf</file>
<file>cally_thigh_left.cmf</file>
<file>cally_thigh_right.cmf</file>
<file>cally_upperarm_left.cmf</file>
<file>cally_upperarm_right.cmf</file>
```

</mesh>

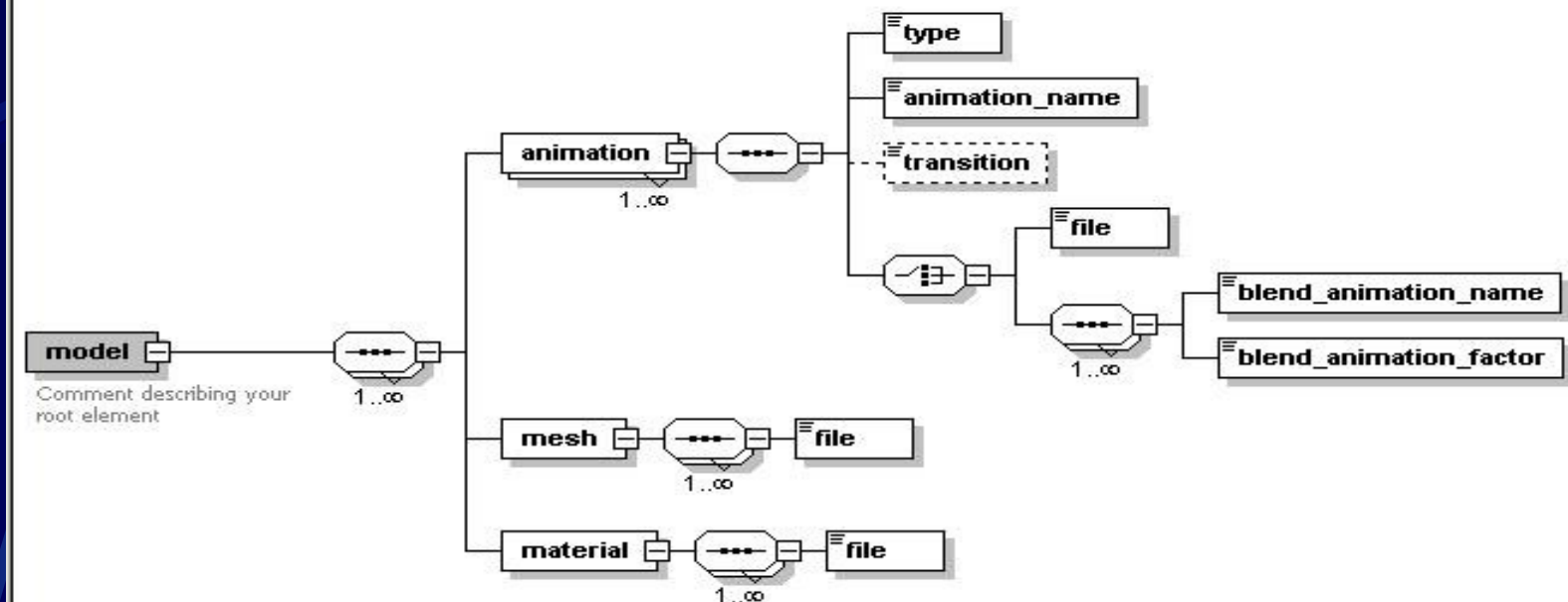
<material>

```
<file>cally_skin.crf</file>
<file>cally_ponytail.crf</file>
<file>cally_chest.crf</file>
<file>cally_pelvis.crf</file>
```

</material>

</model>

# Comparison : The Config File



Attributes					
Identity constraints					
Name	Type	Use	Default	Fixed	
path	xs:string	required			
scale	xs:float	required			
skeleton	xs:string	required			

# Comparison : Parsing the File

```
// find the first non-whitespace character after space
strDataPos = strData.find_first_not_of(" \t", strDataPos + 1);

// get the data
tokenData = strData.substr(strDataPos, strData.find_first_of(" \t", strDataPos) - strDataPos);

//load state transition duration
int trans = atoi(tokenData.c_str());
std::cout << std::endl << "          transition = " << trans << ", ";
m_animations[m_numAnimations].transition = trans;

//move to the white space
strDataPos = strData.find_first_of(" \t", strDataPos + 1);

// find the first non-whitespace character after space
strDataPos = strData.find_first_not_of(" \t", strDataPos + 1);

// get the data
//          endDataPos = strData.find_first_of("\n\r", strDataPos);
tokenData = strData.substr(strDataPos, strData.find_first_of("\n\r", strDataPos) - strDataPos);

// load core animation
std::cout << "file = '" << tokenData << "'..." << std::endl;
m_animations[m_numAnimations].id = m_calCoreModel.loadCoreAnimation(strPath +
tokenData);
if(m_animations[m_numAnimations].id == -1)
```

# Comparison : Parsing the File

```
//while there are more configuration file elements to be processed
while(currentToken < numTokens)
{
    //if we have the model element, first grab its attributes
    if(strcmp("model",xmlArray[currentToken].elementName) == 0)
    {
        currentToken++;
        std::cout << "Initializing model :" << std::endl;

        //get path
        std::cout << xmlArray[currentToken].elementName << " = "
                  << xmlArray[currentToken].elementValue << std::endl;
        path = xmlArray[currentToken++].elementValue;

        //get scale
        std::cout << xmlArray[currentToken].elementName << " = "
                  << xmlArray[currentToken].elementValue << std::endl;
        m_renderScale = atof(xmlArray[currentToken++].elementValue);

        //load skeleton file
        std::string skeletonFile = xmlArray[currentToken++].elementValue;
        std::cout << "Loading skeleton " << skeletonFile << "... " << std::endl;
    }
}
```

# References

- <http://www.alphaworks.ibm.com/tech/xml4c>
- <http://www.w3.org/TR/REC-xml>
- <http://www.sax.sourceforge.net/>